AF/
IFW

IN THE UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS AND INTERFERENCES

Applicant	: Hooper, et. al.	Art Unit	: 2155
Serial No.	: 09/626,535	Examiner	: Eng, David Y
Filed	: 7/27/2000	Assignee	: Intel Corporation
Title	: MULTI-THREADED SCHEDULED RECEIVE FOR FAST NETWORK PORT DATA		

Mail Stop Appeal Brief - Patents
Commissioner for Patents
P.O. Box 1450
Alexandria, VA 22313-1450

APPEAL BRIEF

(i) Real Party in Interest

The real party in interest in this appeal is Intel Corporation, a Delaware corporation having a principal place of business at 2200 Mission College Blvd, Santa Clara, CA 95052. Intel is the assignee of the entire right, title, and interest in the above-noted application.

(ii) Related Appeals and Interferences

An appeal brief for the present case was filed on 8/8/2005. In response to the appeal brief, prosecution was reopened and no decision was rendered by the Board.

A notice of appeal was filed on 4/22/2005 for U.S. serial no. 09/475,614, entitled "Method and Apparatus for Control of Receive Data". The corresponding appeal brief for that application was filed on 7/22/2005. An amended appeal brief was filed 9/25/2006. No decision has been rendered by the Board.

(iii) Status of Claims

Claims 1 and 3-21 are pending and being appealed with claims 1, 7, and 15 being independent.

Claim 2 was previously cancelled.

(iv) Status of Amendments

No amendments were filed after the final rejection mailed on 04/26/2006

(v) Summary of Claimed Subject Matter

The independent claims recite subject matter relating to a processor having multiple programmable multi-threaded engines. For example, FIG. 1 depicts a processor 12 including multiple programmable microengines 22. Each of the microengines provides multiple program threads (page 4, lines 7-9). The processor 12 processes network packets received from one or more media access control devices 30, 31 (page 5, lines 13-page 6, line 2). Different threads provided by the microengines process the network packets (e.g., page 23, lines 7-11). In particular, as recited by the independent claims, different portions/blocks of a given packet are processed by different threads in specific ways.

For example, as recited in claim 1, a second thread is scheduled to process a second incoming block of data within a network packet prior to a first thread completing processing of a first block of data within the packet. For instance, as shown in FIG. 12, thread "y" 304 is scheduled to process a second block of a packet's data before thread "x" 303 completes processing of a first block of the packet's data.

As another example, as recited in claim 7, a second thread can process a second portion of a network packet simultaneously with a first thread processing a first portion of the network packet. For instance, as shown in FIG. 12, while thread "x" 303 processes a first portion of a packet, thread "y" 304 processes a second portion.

Finally, as recited in claim 15, a first thread and a second thread that do not time share can process respective first and second portions of a data packet. For example, threads "x" 303, "y" 304, and "z" 305 can be executed in different microengines but still process different portions of the same packet (page 23, line 24-25). The operation of the threads can be caused by computer-executable instructions executed by the processor (page 6, lines 9-12).

(vi) Grounds of Rejection to be Reviewed on Appeal

Whether claims 1 and 3-21 are unpatentable under 35 U.S.C. 103 over Belkin (6,604,125) in view of Allison (6,373,848).

(vii) Argument

Claims 1, 3, 6, 18, 19, 20

Claim 1 recites a method where a second thread may be scheduled to process a second incoming block of data within a network packet prior to a first thread completing processing of a first block of data within the packet. The Office Action mailed

10/14/2005 rejected claim 1 under 35 U.S.C. 103 as being unpatentable over Belkin (U.S. 6,604,125) in view of Allison (U.S. 6,373,848). In particular, this Office Action equated engines 120-126 of Belkin as the engines recited by claim 1 (Office Action mailed 10/14/2005, page 2). In Belkin, engines 120-126 perform different services for client requests such as providing HTML, CGI, or JAVA services. The threads used to provide these services are assigned from a pool of threads. As described in Belkin:

> the thread pool that is associated with the request is the one that is customized for the type of service being requested... Once the proper thread pool is determined, a thread from that thread pool is assigned to the request. Thereafter, the assigned thread is used to service the request. In servicing the request, the assigned thread and its associated stack are used to execute any computer code that is necessary for servicing the request (Belkin, col. 5, lines 4-10).

Thus, as understood by Applicants, in Belkin, a single engine 120-126 thread services a single request. In other words, this passage does not indicate that multiple engine 120-126 threads operate on the same request in contrast to the subject matter of claim 1 which recites a first thread that operates on a first block of a network packet and a second thread that operates on a second block of the network packet.

In response to this analysis, the Final Office Action mailed 04/26/2006 stated that "the claims do not positively recite that the blocks of data are of the same packet" (Final Office Action mailed 04/26/2006, page 2). Applicants disagree as a matter of claim construction and the proper use of antecedent basis. That is, in claim 1, the first thread processes a block of **a** network packet and the second thread processes a block of **the** network packet. In accordance with proper antecedent basis, "the network packet" refers to the same network packet introduced by the recitation of "a network packet". For these reasons, Applicants respectfully request withdrawal of the rejection.

Claim 4

Dependent claim 4 recites that state information saved by a first thread (i.e., the thread scheduled to process a first incoming block of data within a network packet) and retrieved by the second thread (i.e., the thread scheduled to process a second incoming block of data within the network packet) indicates where to move incoming blocks of data. The Office Action mailed 10/14/2005 rejected claim 4 stating that "the engines of Belkin are inherently capable of saving and retrieving information including information labeled as state information or pointer" (Office Action Mailed 10/14/2005, page 3). This rejection, however, ignores limitations stated by claim 4. That is, assuming for the sake of argument that the mere capability of performing some operation was sufficient to anticipate the operation, the Office Action does not indicate where a teaching of the entire limitation is found or inherently provided. That is, assuming the existence of state information or a pointer does not, inherently, anticipate the specific recitation of "a pointer into a memory indicating where to move the first and second incoming blocks of data". In short, the currently pending rejection did not fully address the claim language of claim 4 or identify a portion of Belkin providing the recitation. Thus, in addition to the reasons discussed in conjunction with claim 1, Applicants respectfully request withdrawal of the claim 4 rejection.

Claim 5

Dependent claim 5 recites storing data to memory in a sequential order based on state information. Neither the Office Action mailed 10/14/2005 nor the Final Office Action mailed 4/26/2006 refers to the limitations of claim 5 or makes any specific argument regarding the unpatentability of the recited subject matter. Thus, in addition to the reasons discussed in conjunction with claims 1 and 4, Applicants respectfully request withdrawal of the claim 5 rejection.

Claims 7, 8, and 9

Independent claim 7 recites a second thread that processes a second portion of a network packet simultaneously with a first thread processing a first portion of the network packet. The Office Action mailed 10/14/2005 rejected claim 7 under 35 U.S.C. 103 as being unpatentable over Belkin (U.S. 6,604,125) in view of Allison (U.S. 6,373,848). In particular, this Office Action equated engines 120-126 of Belkin as the engines recited by claim 7 (Office Action mailed 10/14/2005, page 2). In Belkin, the engines 120-126 perform different services for client requests such as providing HTML, CGI, or JAVA services. The threads used to provide these services are assigned from a pool of threads. As described in Belkin:

> ...the thread pool that is associated with the request is the one that is customized for the type of service being requested... Once the proper thread pool is determined, a thread from that thread pool is assigned to the request. Thereafter, the assigned thread is used to service the request. In servicing the request, the assigned thread and its associated stack are used to execute any computer code that is necessary for servicing the request" (Belkin, col. 5, lines 4-10).

As understood by Applicants, in Belkin, a single engine 120-126 thread services a single request. In other words, this passage does not indicate that multiple engine 120-126 threads operate on the same request in contrast to the subject matter of claim 7 which recites a first thread that operates on a first portion of a network packet and a second thread that simultaneously operates on a second portion of the network packet.

In response to this analysis, the Final Office Action mailed 04/26/2006 stated that "the claims do not positively recite that the blocks of data are of the same packet" (Final Office Action mailed 04/26/2006, page 2). Applicants disagree as a matter of claim construction and the proper use of antecedent basis in claim 7. That is, the first thread processes a first portion of **the** network packet and the second thread processes a second portion of **the** network packet. In accordance with proper antecedent basis, the "the network packet" processed by both threads corresponds to the same network packet introduced by the recitation of "a network packet" introduced earlier in the claim. For these reasons, Applicants respectfully request withdrawal of the rejection of claims 7, 8, and 9.

Claim 10

Dependent claim 10 recites a method where a first thread and second thread time share processing with one another. In rejecting a previous claim, claim 8, the Office Action mailed 10/14/2005 states "there is no time sharing in Belkin". Thus, the rejection of claim 10, which claims time share processing, is inconsistent with the stated position of the Office Action regarding claim 8 which claims no time sharing.

The Office Action mailed 10/14/2005 rejected claim 10 stating that "the wherein clauses merely state the result of the limitations recited in parent claim 7. The clauses therefore add nothing to the patentability or substance of the claims". However, Applicants disagree that claim 10 is a necessary result of claim 7. For example, claim 8 recites "no time sharing" as a possibility while claim 10 recites "time sharing" as another possibility. Thus, in addition to the reasons discussed in conjunction with claim 7, Applicants respectfully request withdrawal of the claim 10 rejection.

Claim 11

Dependent claim 11 recites that the first thread and second thread that time share processing with one another operate out of a common one of the engines. The Office Action mailed 10/14/2005 rejected claim 11 stating that "the wherein clause[s] merely state the result of the limitations recited in parent claim 7. The clauses therefore add nothing to the patentability or substance of the claims". However, Applicants disagree that claim 11 is a necessary result of claim 7. Thus, in addition to the reasons discussed in conjunction with claims 7 and 10, Applicants respectfully request withdrawal of the claim 11 rejection.

Claim 12, 13, and 14

Dependent claim 12 recites processing a third portion of a network packet by a third thread simultaneously with processing of a first and second portion of the network packet. The Office Action mailed 10/14/2005 rejected claim 12 stating "Belkin has more than two engines" (Office Action mailed 10/14/2005, page 3). While Belkin does depict more than two engines 120-126, the Office Action does not state how the existence of more than two engines 120-126 in Belkin teaches or makes inherent the recited subject matter of processing a third portion of a network packet simultaneously with processing of a first and second portion. In the absence of an identification of such a teaching in Belkin, Applicants request withdrawal of the rejection of claim 12, 13, and 14 for these reasons and those discussed in conjunction with claim 7.

Claim 15 and 16

Independent claim 15 recites a first thread and a second thread that do not time share can process respective first and second portions of a data packet. The Office Action mailed 10/14/2005 rejected claim 15 under 35 U.S.C. 103 as being unpatentable over Belkin (U.S. 6,604,125) in view of Allison (U.S. 6,373,848). In particular, this Office Action equated engines 120-126 of Belkin as the engines recited by claim 15 (Office Action mailed 10/14/2005, page 2). In Belkin, the engines 120-126 perform different services for client requests such as providing HTML, CGI, or JAVA services. The threads used to provide these services are assigned from a pool of threads. As described in Belkin:

> ...the thread pool that is associated with the request is the one that is customized for the type of service being requested... Once the proper thread pool is determined, a thread from that thread pool is assigned to the request. Thereafter, the assigned thread is used to service the request. In servicing the request, the assigned thread and its associated stack are used to execute any computer code that is necessary for servicing the request" (Belkin, col. 5, lines 4-10).

As understood by Applicants, in Belkin, a single engine 120-126 thread services a single request. In other words, this passage does not indicate that multiple engine 120-126 threads operate on the same request in contrast to the subject matter of claim 1 which recites a first thread that operates on a first block of a network packet and a second thread that operates on a second block of the network packet.

In response to this analysis, the Final Office Action mailed 04/26/2006 stated that "the claims do not positively recite that the blocks of data are of the same packet" (Final Office Action mailed 04/26/2006, page 2). Applicants disagrees as a matter of claim construction and the proper use of antecedent basis in claim 15. That is, the first thread processes a first portion of **a** data packet and the second thread processes a second portion of **the** data packet. In accordance with proper antecedent basis, the "the data packet" refers to the same network packet introduced by the recitation of "a data packet". For these reasons, Applicants respectfully request withdrawal of the rejections of claims 15 and 16.

Claim 17

Dependent claim 17 recites instructions that provide state information saved by a first thread and retrieved by a second thread (claim 16) to transmit circuitry when an end of packet is detected by a subsequent thread. The Office Action mailed 10/14/2006 rejected claim 17 stating "the engine of Belkin is capable of outputting (inherent) data including state information to a circuitry including circuitry labeling as transmit circuitry" (Office Action mailed 10/14/2006, page 3). Assuming again, purely for the sake of argument, that the mere alleged capability of an apparatus to do something is sufficient to reject a recited element, the Office Action analysis does not address where in Belkin an explicit or inherent description of the claimed language appears. That is, even assuming for the sake of argument that Belkin is capable of outputting data to transmit circuitry and that such capability is sufficient to support a rejection, the Office Action does not identify a teaching of Belkin, inherently or explicitly, indicating this occurs when an end of packet is detected by a subsequent thread. In other words, the Office Action rejection only addressed the beginning of claim 17 instead of the language of the entire claim. Thus, in addition to the reasons discussed in conjunction with claims 15 and 16, Applicants respectfully request withdrawal of the claim 17 rejection.

Claim 20

Dependent claim 20 recites parsing the header of the received network packet, performing a lookup based on the parsing, and enqueuing an entry in a transmit queue for the network packet based on the performed lookup. The Office Action mailed 10/14/2006 rejected claim 20 stating "header parsing is inherent in network communication. The engines of Belknis are capable of performing lookups and enqueuing also". The Office Action did not state that Belkin explicitly or implicitly performs these operations, but that Belkin is capable of doing so. That is, the Office Action does not identify where Belkin actually performs these operations, whether explicitly stated or inherent to the operation of Belkin. Thus, the Office Action fails to identify a teaching in Belkin supporting the Office Action rejection of claim 20. Thus, in addition to the reasons discussed in conjunction with claim 1, Applicant respectfully requests withdrawal of the claim 20 rejection.

Claim 21

Dependent claim 21 recites that each of the multiple programmable multi-threaded engines recited by claim 7 comprises an arithmetic logic unit, a control store, and multiple program counters associated with multiple corresponding threads provided by the engine. The Final Office Action mailed 04/26/2006 rejected claim 21 on the basis that claim 21 was "nothing more than a component list because there is no detail recitation as to how those components are used to process data blocks of the same packet". The Final Office Action did not indicate that the engines 120-126 of Belkin provided these engines either explicitly or inherently. In the absence of an identification of a teaching of such engines in Belkin, Applicants request withdrawal of the rejection of claim 21 for these reasons and those discussed in conjunction with claims 7, 8, and 9.

Please charge the $500.00 brief fee to Deposit Account No. 06-1050. Please charge any additional fees or charges to Deposit Account No. 06-1050.

Respectfully submitted,

Denis G. Maloney
Reg. No. 29,670
On Behalf of Robert A. Greenberg, Esq.

Date: 9/26/2006
/Robert A. Greenberg/
Robert A. Greenberg
Reg. No. 44,133
Phone: 978-553-2060

(viii) Claims Appendix

1. A method of processing network data in a processor having multiple programmable multi-threaded engines integrated within the processor, the method comprising:

scheduling a first thread provided by the multiple programmable multi-threaded engines integrated within the processor to process a first incoming block of data within a network packet received at port of a media access control device ; and

scheduling a second thread provided by the multiple programmable multi-threaded engines integrated within the processor to process a second incoming block of data within the network packet prior to the first thread completing processing of the first incoming block of data.

3. The method of claim 1 further comprising:

saving state information by the first thread; and

retrieving the state information by the second thread.

4. The method of claim 3, wherein the state information includes a pointer into a memory indicating where to move the first and second incoming blocks of data.

5. The method of claim 4 further comprising:

storing data to memory in a sequential ordering based on the state information.

6. The method of claim 5 further comprising:

providing the state information to transmit circuitry.

7. A method of processing a network packet received over a network at a processor having multiple programmable multi-threaded engines integrated within the processor, the method comprising:

processing a first portion of the network packet received at port of a media access control device using a first thread provided by the multiple programmable multi-threaded engines integrated within the processor; and

simultaneously processing a second portion of the network packet using a second thread provided by the multiple programmable multi-threaded engines integrated within the processor.

8. The method of claim 7 wherein the first thread and the second thread do not time share processing with one another.

9. The method of claim 8 wherein the first thread and the second thread operate out of different ones of the multiple multi-threaded engines integrated within the processor.

10. The method of claim 7 wherein the first thread and the second thread time share processing with one another.

11. The method of claim 10 wherein the first thread and the second thread operate out of a common one of the multiple multi-threaded engines integrated within the processor.

12. The method of claim 7 further comprising:

simultaneously with processing the first portion and the second portion of the network packet, processing a third portion of the network packet using a third thread.

13. The method of claim 12 wherein the first thread, the second thread, and the third thread run the same code.

14. The method of claim 13 wherein the first thread, the second thread, and the third thread do not time share processing with one another.

15. An article comprising a computer-readable medium which store computer-executable instructions for receiving data from a plurality of ports, the instructions causing a processor having multiple programmable multi-threaded engines integrated within the processor, the method to:

process a first portion of a data packet using a first thread provided by the multiple programmable multi-threaded engines integrated within the processor; and

process a second portion of the data packet using a second thread provided by the multiple programmable multi-threaded engines integrated within the processor, wherein there is no time sharing between the first thread and the second thread.

16. The article of claim 15, the article further comprises instructions to:

save state information of the first thread; and

restore the state information by the second thread.

17. The article of claim 16, the article further comprises instructions to:

provide the state information to transmit circuitry when an end of packet is detected by a subsequent thread.

18. The method of claim 1, wherein the network packet comprises an Ethernet packet.

19. The method of claim 1, further comprising monitoring the port of the media access control device for received data.

20. The method of claim 1, wherein the processing comprises:

parsing the header of the received network packet;

performing a lookup based on the parsing; and

enqueuing an entry in a transmit queue for the network packet based on the performed lookup.

21, The method of claim 7, wherein each of the multiple programmable multi-threaded engines comprises an arithmetic logic unit, a control store, and multiple program counters associated with multiple corresponding threads provided by the engine.

(ix) Evidence Appendix

None

(x) Related Proceeding Appendix

None